Universidad de Buenos Aires

Facultad de Ingeniería

Módulo electrónico para laboratorio remoto

Especificación de requerimientos de software

Raúl Emilio Romero
rraulemilioromero@gmail.com

15/11/2023
Versión A

Registros de cambios

Revisión	Detalles de los cambios realizados	Fecha
0	Creación del documento	15/11/23

Contents

1 Introducción

1.1 Propósito

1. Este documento representa una especificación de requerimiento de software para un módulo electrónico adquisidor de datos para laboratorios remotos (LR).

2. Está dirigido a desarrolladores que se ocupen del análisis, diseño e implementación, así como también a quienes desarrollen el testing de software.

1.2 Ámbito del sistema

1. Este software forma parte de un desarrollo más amplio llevado a cabo en el contexto de la carrera Maestría en Internet de las Cosas (MIoT) y cuyo objetivo es desarrollar un laboratorio remoto para la enseñanza de la física.

2. Este software llevará el nombre comercial de SACDMELR (Software de adquisición y control de datos para módulos electrónicos de laboratorios remotos).

3. Se comercializará integrado a laboratorios remotos destinados a la enseñanza de la física en carreras de ingeniería.

1.3 Definiciones, Acrónimos y Abreviaturas

API	Application Programming Interface
CEIoT	Carrera de Especialización en Internet de las Cosas
CESE	Carrera de Especialización en Sistemas Embebidos
IoT	Internet de las Cosas
JSON	JavaScript Object Notation
MIoT	Maestría en Internet de las Cosas
MQTT	Message Queuing Telemetry Transport
LR	Laboratorio remoto
TBD	To Be Define
SACDMELR	Software de adquisición y control de datos para módulos electrónicos de laboratorios remotos
UART	Universal Asynchronous Receiver-Transmitter
N/A	No Aplica

1.4 Referencias

1. Plan de Proyecto del Trabajo Final de la Carrera Especialización en Internet de las Cosas. Ing. Raúl Emilio Romero. Plan de Proyecto del Trabajo Final.

1.5 Visión general del documento

1. Este documento se realiza siguiendo el estándar IEEE Std. 830-1998

2 Descripción general del documento

2.1 Perspectiva del producto

El plan de trabajo de la MIoT se separó en dos partes; una dedicada al diseño e implementación general del LR y otra al diseño e implementación de los dispositivos de adquisición y control de datos propios de cada experimento (los requerimientos de software de éstos dispositivos se describen en este documento). En ese contexto, durante el desarrollo de la carrera Especialización en Internet de las Cosas (CEIoT) se desarrolló la primera parte del trabajo, la relacionada con los servicios de software e interfaces de usuario. La segunda, se desarrollará en el marco de la carrera Especialización en Sistemas Embebidos (CESE) y apunta al diseño de los dispositivos de adquisición de datos (denominados módulos electrónicos).

La figura 1 muestra un esquema general de la arquitectura del LR desarrollado y las tecnologías intervinientes. Se separa en niveles que van desde los aspectos directamente vinculados al experimento (dispositivo electromecánico) hasta los que se relacionan con la visualización y control del experimento por el usuario. El dispositivo electromecánico se sitúa en el nivel experimental.

Figure 1: Esquema general de la arquitectura del LR.

El nivel de dispositivos lo integrarán módulos electrónicos que trabajan de forma independiente y que manipularán/registrarán variables físicas del experimento. Cada dispositivo deberá contar con los circuitos y elementos electrónicos necesarios para la manipulación de las variables eléctricas del nivel experimental y con la capacidad de comunicarse con el nivel superior de la arquitectura. Esta comunicación se realizará de a través de internet de forma inalámbrica y por medio del protocolo MQTT.

El nivel de procesamiento/persistencia lo integra una API (Application Programming Interface) que se utiliza para gestionar el flujo de información proveniente de los dispositivos, realizar la persistencia de los datos del experimento y de proporcionar un canal de comunicación con el nivel visualización/control. El nivel de visualización/control cuenta con dos interfaces de usuario, una destinada a usuarios estudiantes y otra a administradores del sistema.

En la figura 2 se muestra un esquema general de los componentes que deberá contar cada módulo electrónico. Cada módulo deberá ser capaz de adquirir señales eléctricas provenientes de sensores analógicos, detectar/manipular señales digitales (como interruptores finales de carrera) y controlar motores de corriente continua o paso a paso de baja potencia. Además, se pretende que la configuración básica de cada dispositivo (credenciales de red, alarmas, escalas, habilitar-deshabilitar canales analógicos) se realice por medio de una terminal serie desde una computadora.

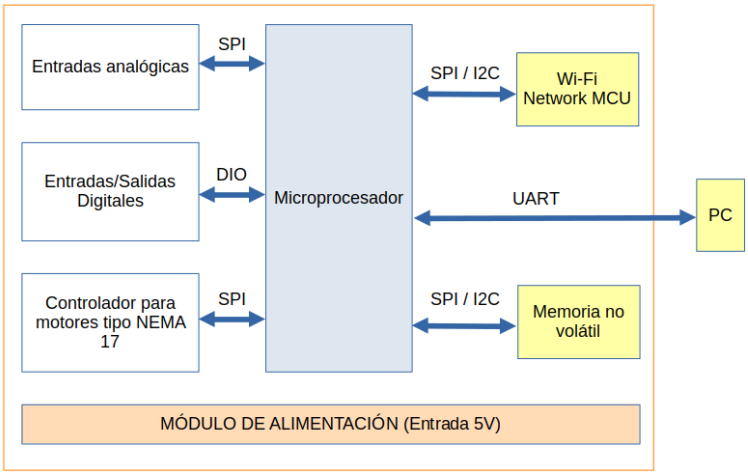

Figure 2: Esquema general del módulo electrónico.

2.2 Funciones del producto

1. El software aquí especificado brindará las siguientes funcionalidades:
   - Adquisición de datos.
   - Control de motores paso-paso de baja potencia.
   - Control de motores de corriente continua de baja potencia.
   - Transmisión y recepción inalámbrica de datos.
   - Comunicación con una PC vía puerto serial.
2. El software aquí especificado no brindará los servicios de:
   - Adquisición y transmisión de datos en tiempo real.

2.3 Características de los usuarios

1. Los usuarios finales de este producto son técnicos electrónicos y profesores de física quienes se ocuparán de la instalación, configuración y mantenimiento del módulo.

2.4 Restricciones

1. El software debe mantenerse bajo control de versiones.
2. El software debe ser escrito en lenguaje C, con las herramientas de desarrollo de la empresa Espressif.

2.5 Suposiciones y dependencias

1. Se presupone que el software correrá en una plataforma de hardware tipo System on Chip de la línea ESP32.

2. Se asume que se dispondrá de la plataforma de desarrollo ESP32-C3-DevKitC-02 durante el tiempo de desarrollo del software para poder debuggear sobre el hardware.

2.6 Requisitos futuros

1. En el futuro se prevé agregar la funcionalidad de que las variables analógicas adquiridas puedan transmitirse en tiempo real.

3 Requisitos específicos

3.1 Interfaces externas

1. Una conexión serie UART. [MELR-ER-0001-REQ0001]

2. Una conexión inalámbrica Wi-Fi. [MELR-ER-0001-REQ0002]

3. Motores de corriente continua. [MELR-ER-0001-REQ0003]

4. Motores paso-paso. [MELR-ER-0001-REQ0004]

5. Sensores de presión y temperatura. [MELR-ER-0001-REQ0005]

6. Sensores de posición. [MELR-ER-0001-REQ0006]

7. Sensores inductivos. [MELR-ER-0001-REQ0007]

8. Sondas de tensión. [MELR-ER-0001-REQ0008]

9. Interruptores finales de carrera. [MELR-ER-0001-REQ0009]

3.2 Funciones

3.2.1 Configuración

1. El software permitirá recibir mensajes desde la terminal serial solo si el interruptor de configuración esta en estado ON (nivel lógico TTL alto). Por el contrario, si el nivel lógico es bajo el software solo responderá a eventos MQTT externos. [MELR-ER-0001-REQ0010]

2. El software recibirá vía interfaz serie UART un identificar de dispositivo (ID), almacenará el número en una memoria no volátil, y responderá por medio de un mensaje vía UART que muestre el ID almacenado. [MELR-ER-0001-REQ0011]

3. El software recibirá vía interfaz serie UART las credenciales de una red Wi-Fi, almacenará la configuración en una memoria no volátil, realizará el proceso de conexión a la red y responderá por medio de un mensaje vía UART si pudo conectarse o no a la red. [MELR-ER-0001-REQ0012]

4. El software recibirá vía interfaz serie UART la frecuencia máxima de muestreo por canal analógico a setear, almacenará la configuración en una memoria no volátil y responderá con un mensaje vía UART si la configuración pudo realizarse o no. [MELR-ER-0001-REQ0013]

5. El software recibirá vía interfaz serie UART un mensaje para activar/desactivar el manejo de motores, almacenará la configuración en una memoria no volátil y responderá con un mensaje vía UART si la configuración pudo realizarse o no. [MELR-ER-0001-REQ0014]

6. El software recibirá vía interfaz serie UART un mensaje para seleccionar el tipo de motor a controlar (motor de continua o motor paso-paso), almacenará la configuración en una memoria no volátil y responderá con un mensaje vía UART si la configuración pudo realizarse o no. [MELR-ER-0001-REQ0015]

3.2.2 Adquisición y control de datos

1. El software recibirá desde la API del LR un mensaje en formato JSON de acuse de conexión por medio MQTT que deberá responder en el mismo formato con un único campo, ID de dispositivo. [MELR-ER-0001-REQ0016]

2. El software recibirá desde la API del LR un mensaje en formato JSON con la solicitud de adquirir datos analógicos (cantidad de canales y tiempo de registro) y responderá por medio de un mensaje en el mismo formato con los datos adquiridos. [MELR-ER-0001-REQ0017]

3. El software deberá encender o apagar un motor a partir de un mensaje recibido por MQTT en formato JSON desde la API del LR. [MELR-ER-0001-REQ0018]

4. El software deberá cambiar la velocidad (entre dos opciones) de un motor a partir de un mensaje recibido desde MQTT en formato JSON desde la API del LR. [MELR-ER-0001-REQ0019]

5. El software recibirá desde la API del LR un mensaje en formato JSON con la solicitud de los estados de las entradas digitales (estado alto o bajo) y responderá con otro mensaje en el mismo formato con los estados de las entradas digitales. [MELR-ER-0001-REQ0020]

6. El software recibirá desde la API del LR un mensaje en formato JSON con el seteo de los estados de las salidas digitales (estado alto o bajo) y realizará el seteo sobre las salidas. [MELR-ER-0001-REQ0021]

3.2.3 Conectividad

1. El software recibirá desde la API del LR un mensaje en formato JSON de solicitud de configuración completa que deberá responder en el mismo formato con los campos: ID, credenciales, frecuencia de muestreo y controlador de motor activado/desactivado). [MELR-ER-0001-REQ0022]

2. El software deberá enviar cada 30 min a la API del LR un mensaje en formato JSON con la siguiente información: ID, nivel de señal Wi-Fi y timestamp). [MELR-ER-0001-REQ0023]

3.2.4 Visualización

1. El software encenderá un led de color verde (de forma continua) si esta conectado a una red de Wi-Fi. La condición de no conectado se indicará con un led rojo en modo intermitente (1 segundo de ciclo) y led verde apagado. [MELR-ER-0001-REQ0024]

2. El software encenderá un led de color verde (de forma intermitente durante 3 segundos) cada vez que envíe o reciba un mensaje desde la API de LR. [MELR-ER-0001-REQ0025]

3. El software encenderá un led de color amarillo en modo continuo si el consumo de corriente del módulo electrónico sobrepasa 1 A. Si la corriente es menor a 1 A el led se apagará. [MELR-ER-0001-REQ0026]

3.3 Requisitos de rendimiento

1. Los datos adquiridos de las variables analógicas deberán ser enviados por MQTT al servidor del LR en un tiempo menor a 1000 ms. [MELR-ER-0001-REQ0027]

3.4 Restricciones de diseño

1. Se utilizará el microcontrolador ESP32-C3 como dispositivo principal. [MELR-ER-0001-REQ0028]

3.5 Atributos del sistema

3.5.1 Mantenibilidad

1. El software debe permitir modificar sus parámetros operativos y almacenarlos en memoria segura. [MELR-ER-0001-REQ0029]

3.5.2 Confiabilidad

1. El software debe asegurar su correcto funcionamiento en condiciones normales de operación durante al menos 1 semana de uso continuo (sin ser reiniciado). [MELR-ER-0001-REQ0020]

3.6 Otros requisitos

N/A.

4 Apéndices

N/A.